Implementación del algoritmo de Deutsch y Deutsch-Jozsa

Jaime Nicolás Castro Acuña
Escuela Colombiana de Ingeniería Julio Garavito
jaime.cacuna@mail.escuelaing.edu.co

27 de noviembre de 2022

Este reporte se entrega para cumplir con los requisitos parciales del curso CNYT: Computación Cuántica- 2022-2

Tabla de contenidos

1 Introducción

Para dar una definición completa sobre la computación cuántica, definiremos algunos conceptos clave a continuación:

Vamos a comenzar con la definición de cuántica la cual hace referencia a la mecánica cuántica que un sistema usa para el cálculo de distintos resultados. En relación con la física, un quantum es la unidad más pequeña en cualquier propiedad física. Comúnmente se hace referencia a las propiedades de partículas atómicas, como lo son neutrinos, fotones y electrones. Ahora continuaremos con la definición de un qbit la cuál es la unidad más básica de información en la computación cuántica. Los qbits en la computación cuántica juegan un papel similar al rol de los bits en la computación clásica, sin embargo, estos se comportan de forma diferente. Los bits clásicos pueden tener representaciones de 0's y 1's, mientras que los qbits pueden tener una superposición de cualquier estado posible. Ahora si para terminar con esta breve introducción podemos definir la computación cuántica de la siguiente manera: Los equipos cuánticos usan el comportamiento singular de la física cuántica, como la superposición y la interferencia cuántica, y lo aplican al cálculo. Para poder representar un qbit debemos representarlo como la combinación lineal de los estados 0 y 1, así:

$$|0 > = [\ 1,0]^T$$

$$|1 > = [\ 0,1]^T$$

Por lo tanto, la representación de un qbit será la siguiente, con c1 y c2 como números complejos:

$$|\psi > = c_1|0 > \ + c_2|1 >$$

Hasta el momento es IBM quien ha hecho su mayor esfuerzo por construir supercomputadoras capaces de resolver problemas con una "alta complejidad". El sistema de estos computadores cuánticos, se utilizan en algunos carros eléctricos (como los de la marca Mercedes-Benz), también para resolver algunas interrogantes cósmicas y físicas o resolver algunos desafíos energéticos complejos, entre otros usos.
Los ordenadores cuánticos, aún en procesos de experimentación, están basados como lo decíamos anteriormente en qbits, pero también usan fotones, es decir, las partículas de la luz. Actualmente, científicos han dado nuevos pasos en este último modelo al lograr en tan solo 36 microsegundos hacer una tarea que los clásicos tardarían casi 9.000 años. Este gran logro se le atribuye a un procesador fotónico cuántico, llamado Borealis.
A continuación, encontrará el problema del Algoritmo de Deutsch junto con las implementaciones de las funciones en el computador cuántico. Además, encontrará la implementación del algoritmo de Deutsch en un computador cuántico. En la tercera parte del documento encontrará la definición del problema del Algoritmo de Deutsch-Jozsa, su implementación de las funciones en el computador cuántico. Además, encontrará la

implementación del algoritmo de Deutsch-Jozsa en un computador cuántico. Al final las conclusiones respectivas de lo encontrado anteriormente.

2 Algoritmo de Deutsch

El experimento que se va a realizar a continuación determina si una función mostrada como una caja negra es constante o es balanceada, así:

$$\mathrm{f}(0) = \mathrm{f}(1) \text{ o } \mathrm{f}(0) \neq \mathrm{f}(1)$$

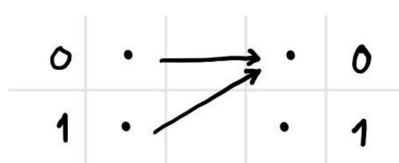
las funciones representadas anteriormente corresponden directamente con el tipo de función asociada. Se pretende con el algoritmo tomar dos bits y colocarlos en un estado de superposición y determinar el estado del bit, si es |0> la función dada es constante, si no, es balanceada.

2.1 Problema

El problema que resuelve el algoritmo de Deutsch es que nos permite determinar cuándo una función es balanceada o constante de forma más eficiente, por medio de la superposición de compuertas clásicas en este caso la compuerta de Hadamard. El experimento que se va a realizar a continuación describe cómo es el comportamiento de las 4 funciones posibles de {0,1} a {0,1}. El simulador cuántico de IBM usado fue https://quantumcomputing.ibm.com/composer

2.2 Implementando las funciones en el computador cuántico

Función # 1:



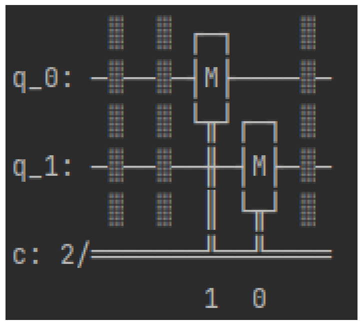
La función anteriormente descrita podemos decir que es contante gracias a lo siguiente:

$$\mathrm{f}(0) = \mathrm{f}(1) = 0$$

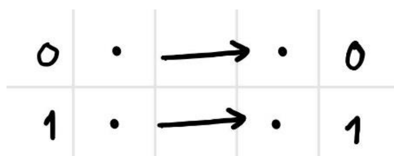
De acuerdo a la función anterior podemos representar esta en forma matricial de la siguiente manera:

```
0--->0
1--->0
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
```

Para que dicho sea este el circuito de las funciones, este debe ser el inverso del mismo, es decir si se aplica dos veces el mismo circuito, la entrada y la salida deben ser la misma, puesto que se anulan entre sí.



Función # 2:



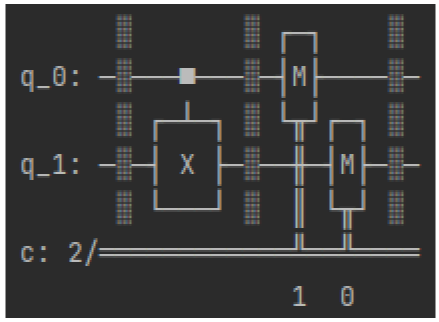
La función anteriormente descrita podemos decir que es contante gracias a lo siguiente:

$$f(0) = 0 \neq f(1) = 1$$

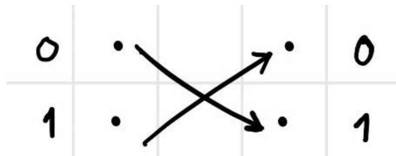
De acuerdo a la función anterior podemos representar esta en forma matricial de la siguiente manera:

```
0--->0
1--->1
1 0 0 0
0 1 0 0
0 0 0 1
0 0 1 0
```

Al utilizar sobre dos bits, cuando el bit de arriba es 1 el mismo se verá afectado invirtiendo el estado, así:



Función # 3:



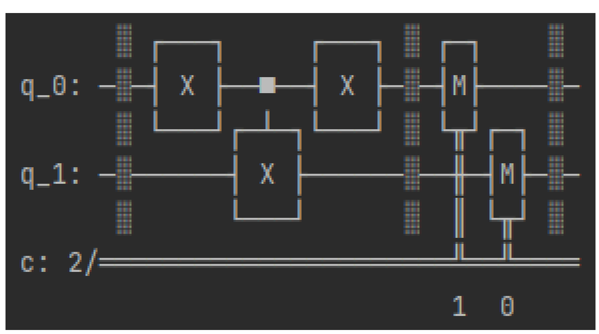
La función anteriormente descrita podemos decir que es contante gracias a lo siguiente:

$$f(0) = 1 \neq f(1) = 0$$

De acuerdo a la función anterior podemos representar esta en forma matricial de la siguiente manera:

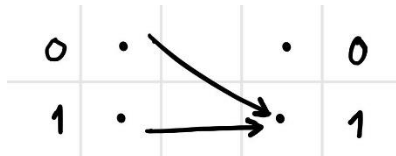
```
0--->1
1--->0
0 1 0 0
1 0 0 0
0 0 1 0
0 0 0 1
```

Como podemos evidenciar esta función realiza todo lo contrario a lo evidenciado en la función anteriormente descrita, en vez de realizar un cambio en el bit inferior cuando el bit superior es 1, realiza el cambio cuando el bit superior es 0, podemos ver la representación de la función en el siguiente circuito:



Haciendo un pequeño análisis de lo sucedido en el circuito anterior vemos que se aplica 2 veces la compuerta NOT en el bit superior y también se aplica la misma compuerta NOT ahora controlado en medio de los dos bits.

Función # 4:



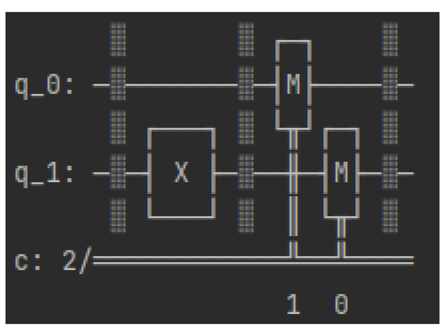
La función anteriormente descrita podemos decir que es contante gracias a lo siguiente:

$$f(0) = f(1) = 1$$

De acuerdo a la función anterior podemos representar esta en forma matricial de la siguiente manera:

```
0--->1
1--->1
0 1 0 0
1 0 0 0
0 0 0 1
0 0 1 0
```

La función número 4 descrita tiene la particularidad de que sin importar el estado del bit superior siempre cambia el estado del bit inferior, lo anterior lo vemos representado a detalle en la siguiente imagen del circuto:



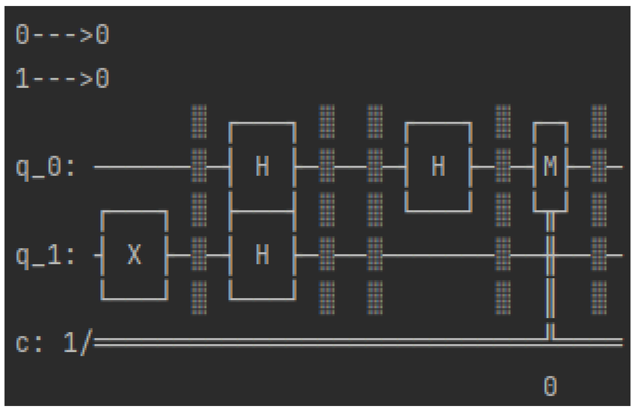
2.3 Implementando el algoritmo de Deutsch en un computador cuántico

Una vez terminado el apartado anterior donde encontramos los circuitos y matrices relacionadas a las funciones dadas. En este aparatado vamos a realizar la simulación de las funciones previamente evaluadas con el algoritmo de Deutsch comprobando su correcto funcionamiento.

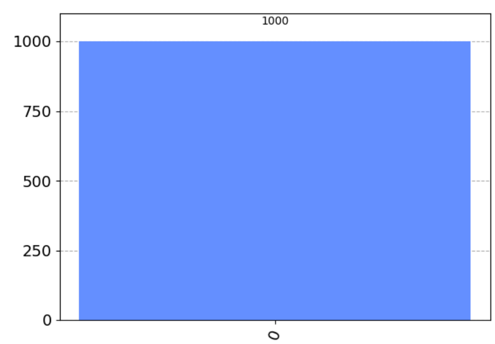
Para esto vamos a ilustrar los circuitos en el mismo orden que se mostraron en el apartado 2.2. Si desea puede correr el archivo “AlgDeutsch.py” para ver la construcción de los circuitos seguidamente mostrados:

Función # 1:

El circuito que describe a la primera función aplicada con el algoritmo de Deutsch es:

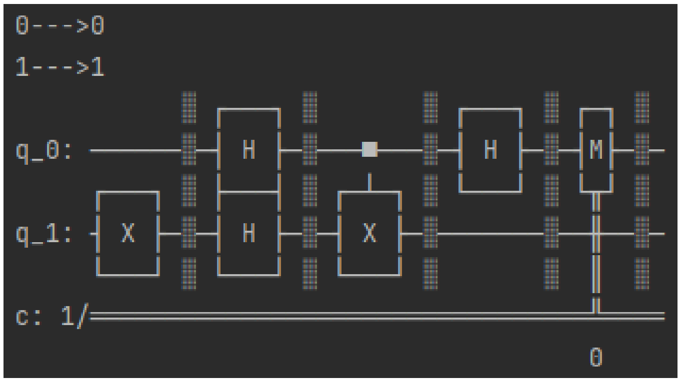


Analizando la imagen anteriormente descrita vemos la función aplicada en forma de circuito que se encuentra entre las matrices de Hadamard. En este caso obtenemos el estado del bit superior = |0>, por lo que se puede concluir que la función es constante.

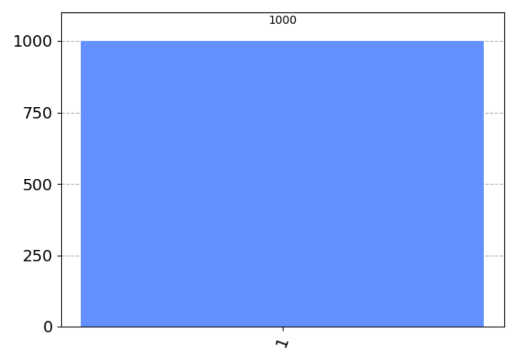


Función # 2:

El circuito que describe a la segunda función aplicada con el algoritmo de Deutsch es:

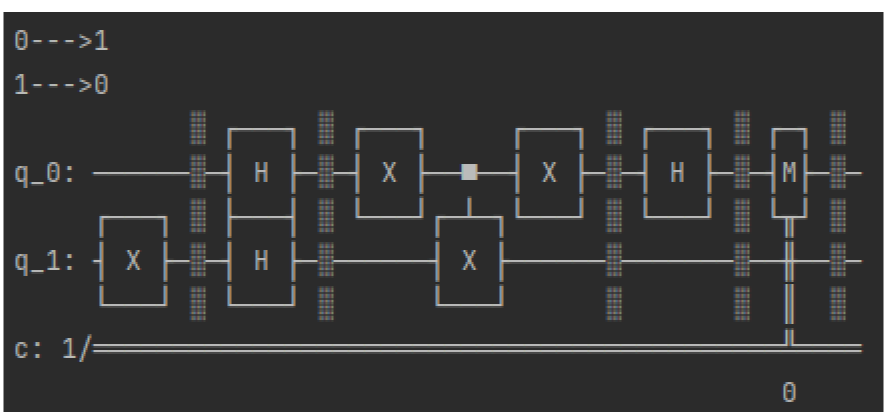


Como mencionamos en el análisis anterior vemos la función aplicada en forma de circuito que se encuentra entre las matrices de Hadamard. En este caso obtenemos el estado del bit superior = |1>, por lo que se puede concluir que la función es balanceada. Revisamos la gráfica obtenida:

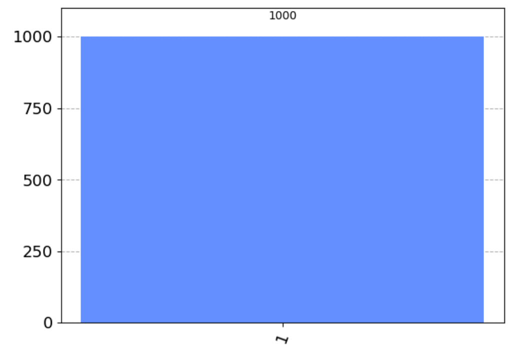


Función # 3:

El circuito que describe a la tercera función aplicada con el algoritmo de Deutsch es:

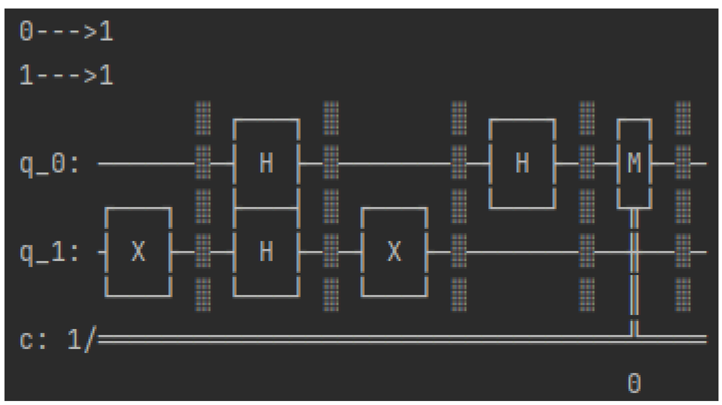


Como mencionamos en el análisis anterior vemos la función aplicada en forma de circuito que se encuentra entre las matrices de Hadamard. En este caso obtenemos el estado del bit superior = |1>, por lo que se puede concluir que la función es balanceada. Revisamos la gráfica obtenida:

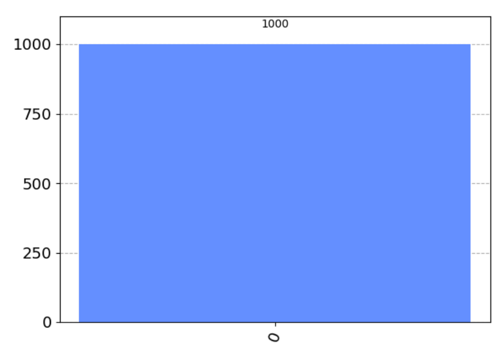


Función # 4:

El circuito que describe a la cuarta función aplicada con el algoritmo de Deutsch es:



Analizando la imagen anteriormente descrita vemos la función aplicada en forma de circuito que se encuentra entre las matrices de Hadamard. En este caso obtenemos el estado del bit superior = |0>, por lo que se puede concluir que la función es constante.



3 Algoritmo de Deutsch-Jozsa

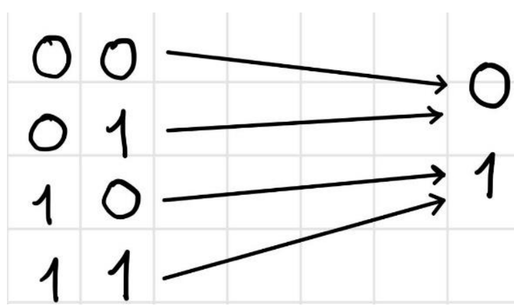
El Deutsch-Jozsa nos ayuda a determinar si una función desconocida dada es balanceada o constante, en general se resuelve el mismo problema que con el algoritmo de Deutsch, generalizando el dominio a n, lo que permite la entrada de cadenas binarias con n bits.

3.1 Problema

Como el algoritmo de Deutsch, el algoritmo de Deutsch-Jozsa, permite saber si una función es balanceada o constante, sin embargo, el algoritmo de Deutsch-Jozsa es la generalización del algoritmo de Deutsch, es decir este algoritmo no solo sirve para un qubit, sino que sirve para n qubits. En este reporte se hará estudio de dos funciones de 2 qubits, una balanceada y la otra constante.

Sabemos que Deutsch-Jozsa posee un dominio representado de la siguiente manera: $\{0,1\}^n$ y un rango por su parte igual a la función de Deutsch $\{0,1\}$. El algoritmo de Deutsch tiene como parámetro las funciones $\{0,1\}^n \rightarrow \{0,1\}$ que cumplen tal condición.

En este caso podemos decir que, existen 2^{2^n} funciones posibles, donde 2 de ellas son constantes cuando todas las entradas se dirigen a 1 o 0 y son $C_{2^n}^{2^{n-1}}$ la cantidad de funciones balanceadas. A continuación, vemos un ejemplo donde n = 2:

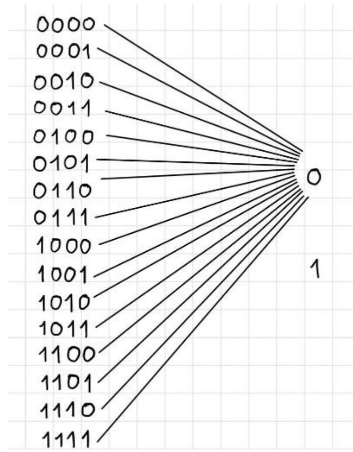


3.2 Implementando las funciones en el computador cuántico

Debido a la explicación anteriormente propuesta vemos que el algoritmo de Deutsch-Jozsa tiene una magnitud demasiado compleja por lo que no se puede abarcar en totalidad. Por tanto, se va a trabajar con un n=4 y de estas funciones tomadas 1 será constante y 3 de ellas serán balanceadas. Todo esto se simulará en Python con la ayuda de la librería qiskit.

Función # 1:

La primera función es la función constante en la cual todos sus valores se dirigen a 0, su representación es:

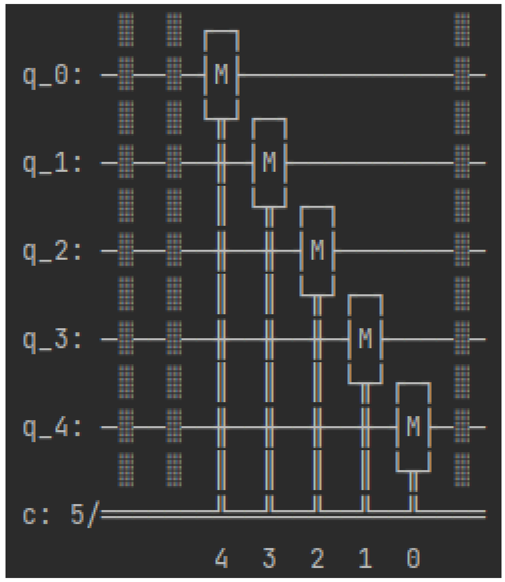


La representación de la función anteriormente descrita en forma de matriz es la siguiente:

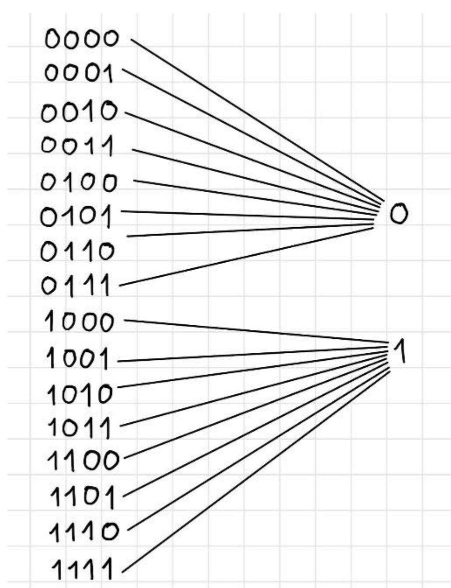
```
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
```

Como podemos observar en la gráfica de la matriz anteriormente descrita la función no realizará ninguna modificación a ninguno de los bits. Lo que nos quiere decir que sin importar el valor que se ingrese siempre se retornará el mismo valor de la función.

Describimos la función a través del circuito que se muestra a continuación:



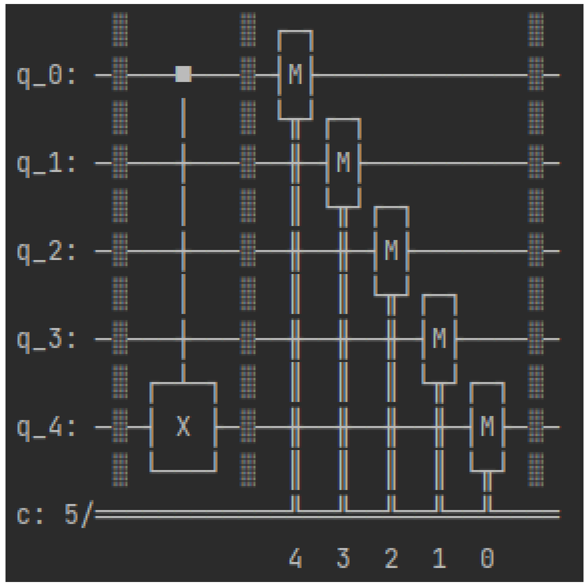
Función # 2:
La segunda función que vamos a representar corresponde a una función balanceada, lo que nos indica que si el primer bit tiene el valor de 1, el último bit cambiará su estado, la representación de la función es la siguiente:



La representación de la función anteriormente descrita en forma de matriz es la siguiente:

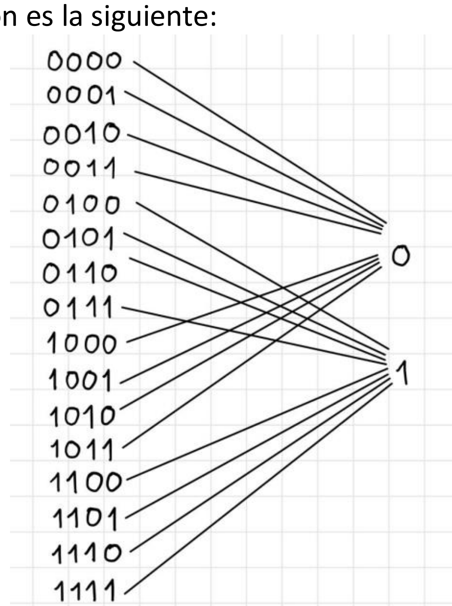
```
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
```

Describimos la función a través del circuito que se muestra a continuación:



Función # 3:

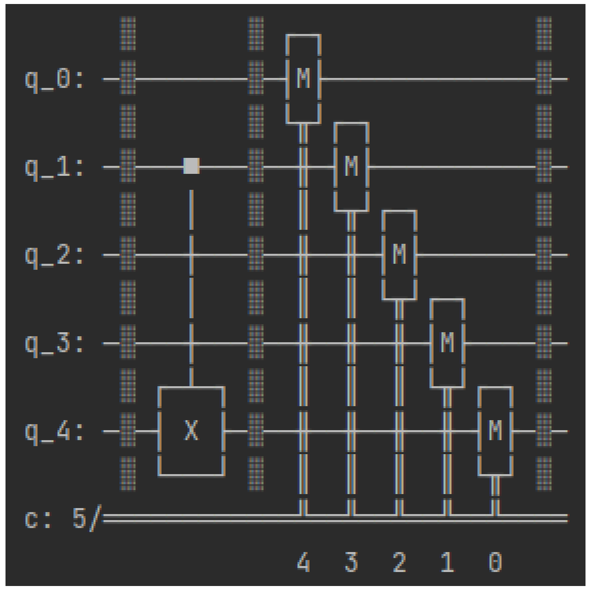
La tercera función que vamos a representar corresponde a una función balanceada, lo que nos indica que, si el segundo bit tiene el valor de 1, el último bit cambiará su estado, la representación de la función es la siguiente:



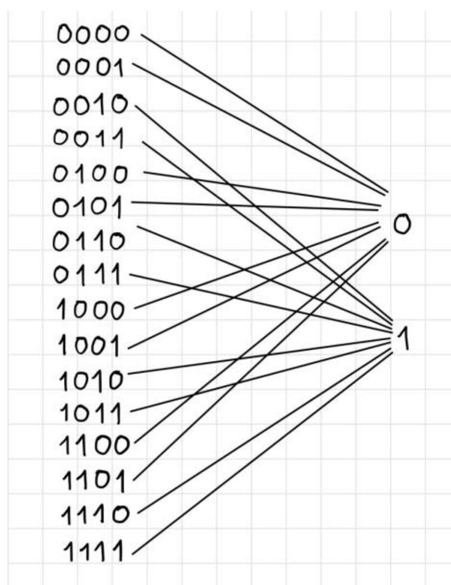
La representación de la función anteriormente descrita en forma de matriz es la siguiente:

```
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
```

Describimos la función a través del circuito que se muestra a continuación:



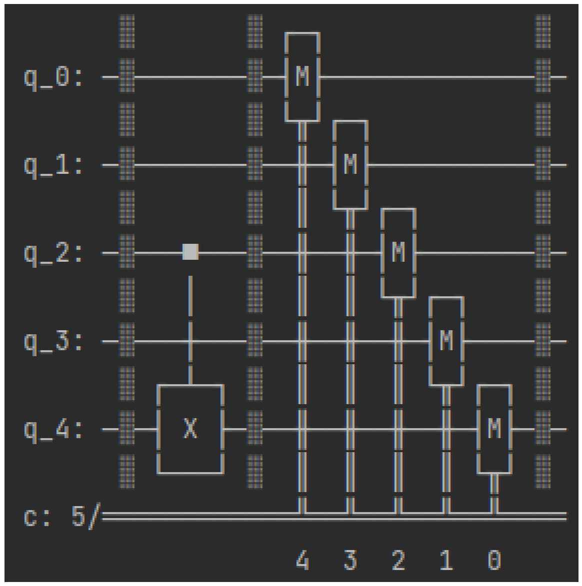
Función # 4:
La tercera función que vamos a representar corresponde a una función balanceada, lo que nos indica que, si el tercer bit tiene el valor de 1, el último bit cambiará su estado, la representación de la función es la siguiente:



La representación de la función anteriormente descrita en forma de matriz es la siguiente:

```
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
```

Describimos la función a través del circuito que se muestra a continuación:

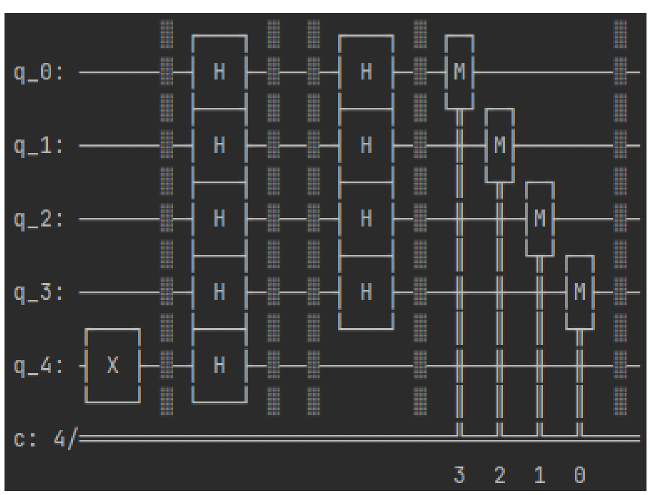


3.3 Implementando el algoritmo de Deutsch-Josza en un computador cuántico

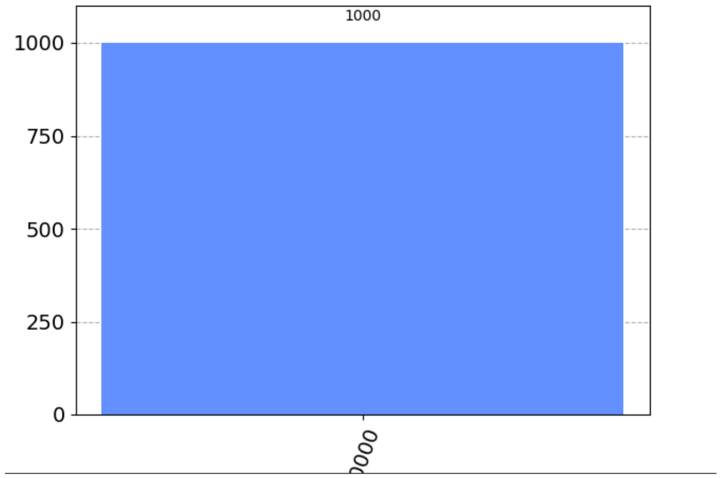
Una vez terminado el apartado anterior donde encontramos los circuitos y matrices relacionadas a las funciones dadas. En este aparatado vamos a realizar la simulación de las funciones previamente evaluadas con el algoritmo de Deutsch-Jozsa comprobando su correcto funcionamiento.

Para esto vamos a ilustrar los circuitos en el mismo orden que se mostraron en el apartado 3.2. Si desea puede correr el archivo “AlgDeutsch-jozsa.py” para ver la construcción de los circuitos seguidamente mostrados:

Función # 1:
El circuito que describe la primera función dada junto al algoritmo de Deutsch-Jozsa tiene la siguiente representación:

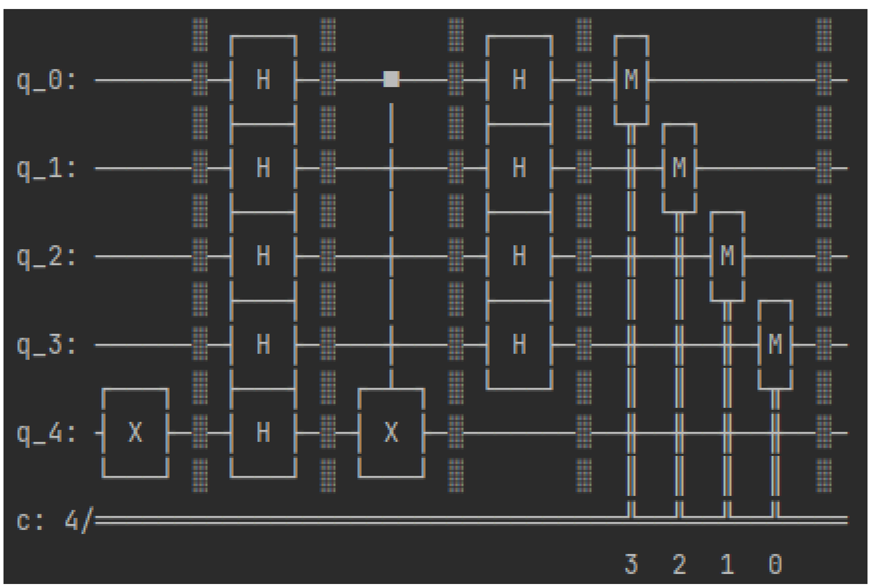


A través de la gráfica obtenida de la función dada # 1 observamos que esta es constante:

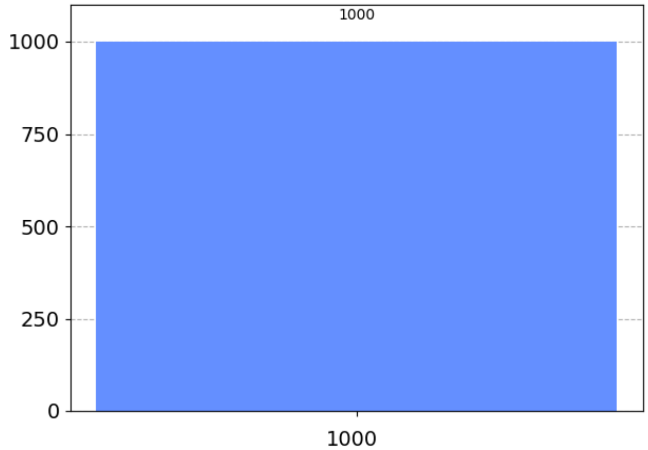


Función # 2:

El circuito que describe la segunda función dada junto al algoritmo de Deutsch-Jozsa tiene la siguiente representación:

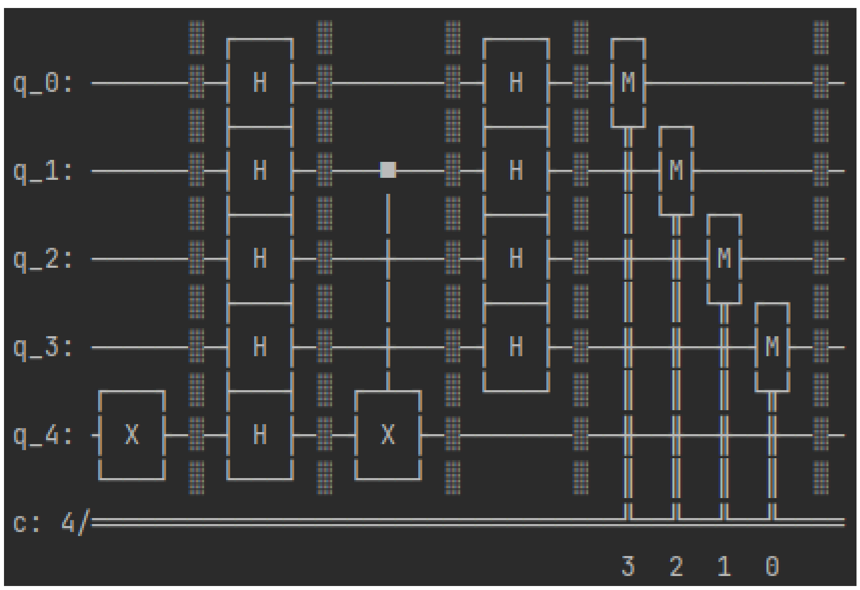


A través de la gráfica obtenida de la función dada # 2 observamos que esta es balanceada:

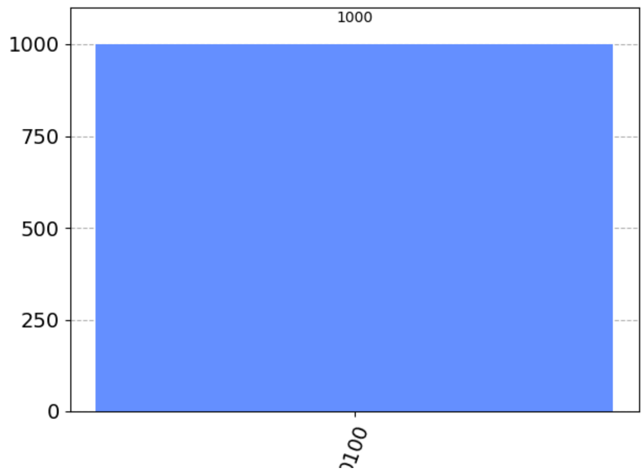


Función # 3:

El circuito que describe la tercera función dada junto al algoritmo de Deutsch-Jozsa tiene la siguiente representación:

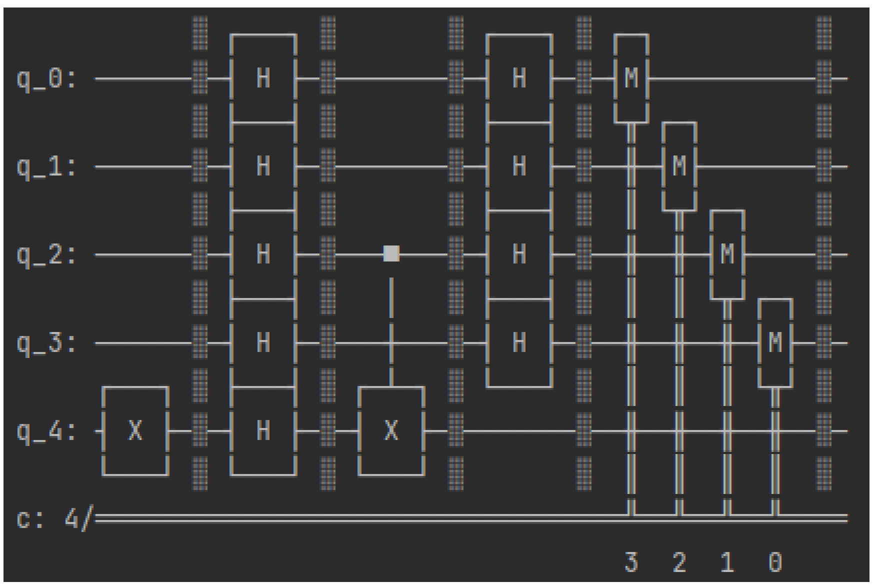


A través de la gráfica obtenida de la función dada # 3 observamos que esta es balanceada:

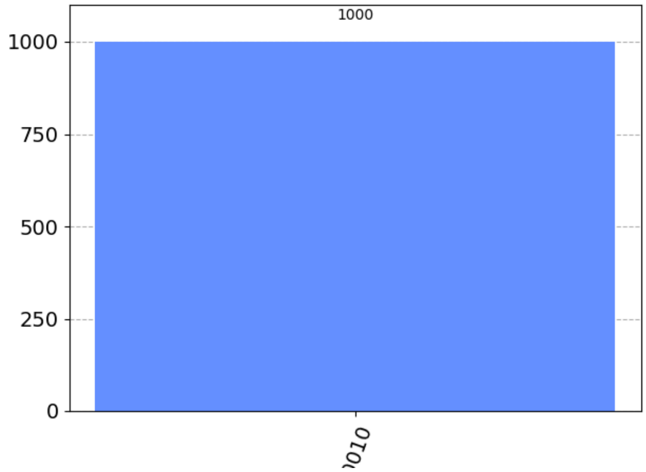


Función # 4:

El circuito que describe la cuarta función dada junto al algoritmo de Deutsch-Jozsa tiene la siguiente representación:



A través de la gráfica obtenida de la función dada # 4 observamos que esta es balanceada:



4 Conclusiones

Con la realización del presente informe pudimos ver inicialmente como es el funcionamiento de las 4 funciones posibles de {0,1} a {0,1} y como estas se ven representadas usando el computador cuántico de IBM. Lo anteriormente realizado nos dio claridad en el funcionamiento del algoritmo de Deutsch, para posteriormente realizar su implementación.
Luego de esto pudimos explicar funciones con rango y dominio {0, 1}^n a {0, 1} respectivamente, representadas posteriormente con compuertas cuánticas. Este tipo de funciones nos ayudaron igualmente que en Deutsch en la implementación del algoritmo de Deutsch-Jozsa. Todo esto con el fin de analizar cuales de las funciones representadas eran balanceadas y cuáles no.
A través de la realización del informe nos dimos de cuenta que la optimización del tiempo era enorme, puesto que si intentáramos implementar alguno de los dos algoritmos previamente descritos debíamos llamar a la función por lo menos 2^{n-1}veces, mientras que con el uso del algoritmo cuántico solo debemos llamar la función una sola vez que los qbits estén pasando por la compuerta Uf.

Se plantea para trabajos posteriores hacer un estudio un poco más detallado acerca del funcionamiento de los algoritmos que ejecutamos sobre las computadoras cuánticas, esto con el fin de no solo ver resultados, sino que también podamos entender y atraer a un mayor número de personas al amplio ecosistema de hardware de la computación cuántica que existe hoy en día.

5 Bibliografía

Qiskit 0.39.3 documentation. Qiskit Development Team. Disponible en: https://acortar.link/HMcrwGhttps://qiskit.org/documentation/ (Consultado: Noviembre 27, 2022).

Getting started - Qiskit 0.39.3 documentation (2022). Qiskit Development Team. Disponible en: https://qiskit.org/documentation/getting_started.html (Consultado: Noviembre 27, 2022).

Welle, D. (2022) *Ordenador Cuántico Logra en 36 microsegundos lo que uno clásico en 9.000 años: DW: 22.06.2022*, *DW.COM*. Made for minds. Disponible en: https://acortar.link/HMcrwG. (Consultado: Noviembre 27, 2022).

Iberdrola (2021) *La Computación Cuántica y las supercomputadoras que revolucionarán La Tecnología*, *Iberdrola*. Iberdrola. Disponible en: https://www.iberdrola.com/innovacion/que-es-computacion-cuantica (Consultado: Noviembre 27, 2022).

https://www.docirs.cl/algoritmo_de_deutsch.asp (2020) *Algoritmo de Deutsch ~ Conceptos Matemticos Bsicos de computacin cuntica*. Document Information Retrieval Systems Technology. Disponible en: https://www.docirs.cl/algoritmo_de_deutsch.asp (Consultado: Noviembre 27, 2022).

Qué Es la Computación Cuántica: Microsoft azure (2022) *Qué es la computación cuántica*. Microsoft Azure Team. Disponible en: https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-quantum-computing/#introduction (Consultado: Noviembre 27, 2022).